FIG. 1 PRIOR ART

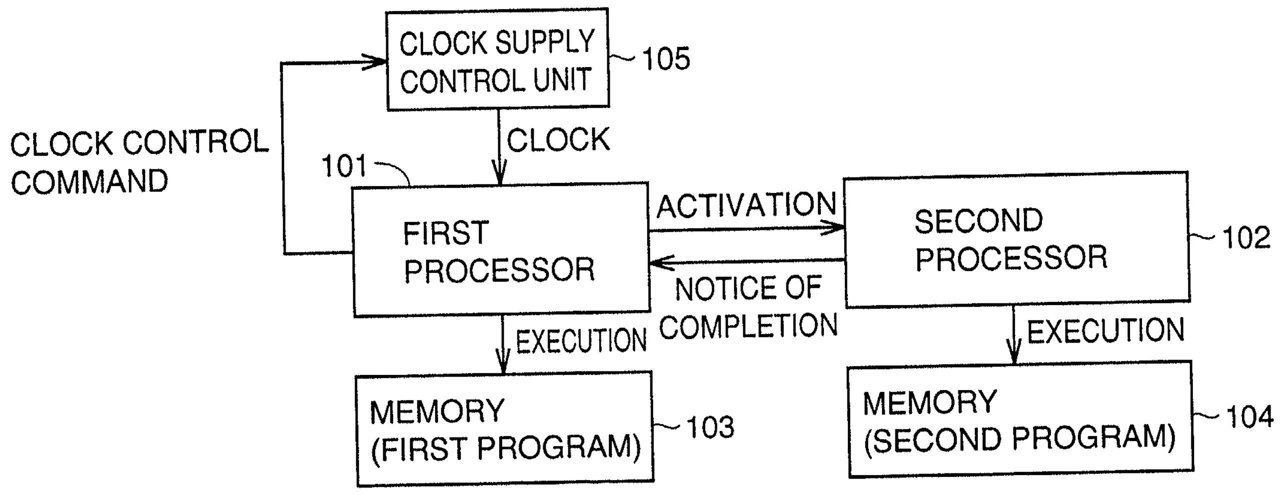


FIG. 2

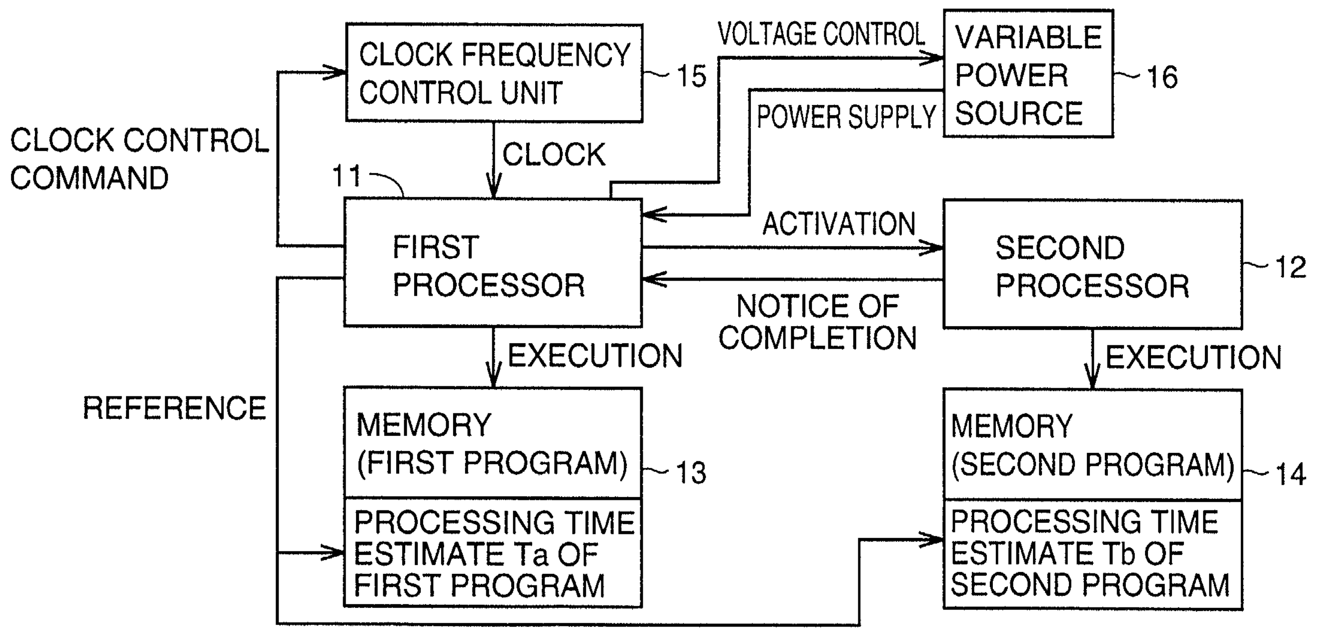

CLOCK FREQUENCY CONTROL UNIT
15
VOLTAGE CONTROL
VARIABLE POWER SOURCE
16
POWER SUPPLY
CLOCK CONTROL COMMAND
CLOCK
11
FIRST PROCESSOR
ACTIVATION
SECOND PROCESSOR
12
NOTICE OF COMPLETION
EXECUTION
EXECUTION
REFERENCE
MEMORY (FIRST PROGRAM)
13
PROCESSING TIME ESTIMATE Ta OF FIRST PROGRAM
MEMORY (SECOND PROGRAM)
14
PROCESSING TIME ESTIMATE Tb OF SECOND PROGRAM

FIG. 3

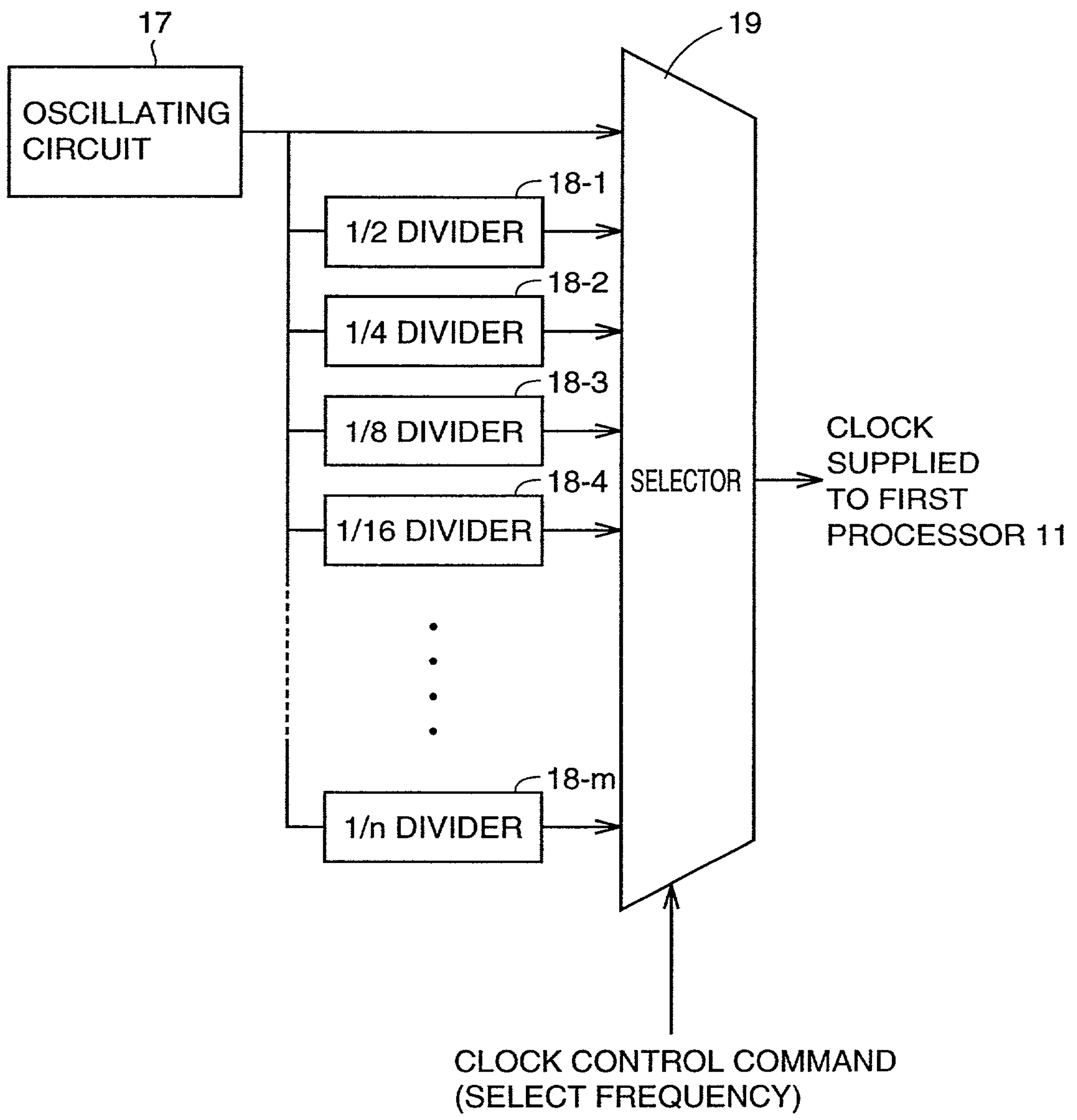
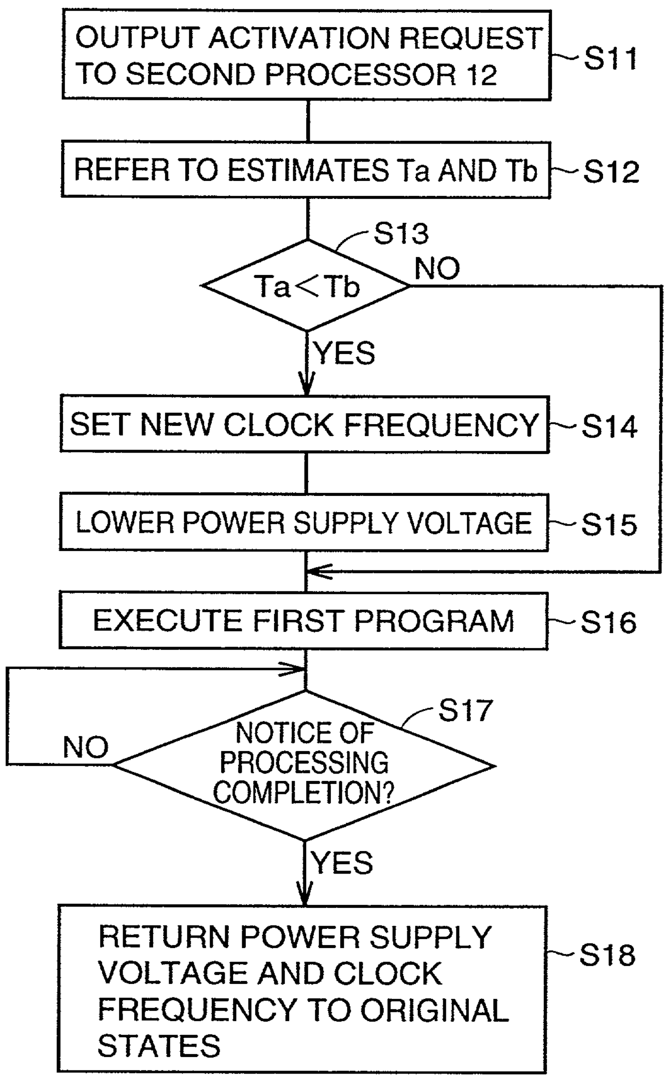

17
OSCILLATING CIRCUIT
19
18-1
1/2 DIVIDER
18-2
1/4 DIVIDER
18-3
1/8 DIVIDER
18-4
1/16 DIVIDER
18-m
1/n DIVIDER
SELECTOR
CLOCK SUPPLIED TO FIRST PROCESSOR 11
CLOCK CONTROL COMMAND (SELECT FREQUENCY)

FIG. 4



OUTPUT ACTIVATION REQUEST TO SECOND PROCESSOR 12
S11
REFER TO ESTIMATES Ta AND Tb
S12
S13
Ta<Tb
NO
YES
SET NEW CLOCK FREQUENCY
S14
LOWER POWER SUPPLY VOLTAGE
S15
EXECUTE FIRST PROGRAM
S16
S17
NOTICE OF PROCESSING COMPLETION?
NO
YES
RETURN POWER SUPPLY VOLTAGE AND CLOCK FREQUENCY TO ORIGINAL STATES
S18

FIG. 5

```
main_procedure()                /* MAIN PROCESSING UNIT OF FIRST PROGRAM */
{

 B_is_done=0;
 invokeB();                     /* ACTIVATE SECOND PROGRAM.  PROCESSING TIME ESTIMATE IS Tb */

 set_clock_frequency(Ta/Tb);    /* LOWER CLOCK FREQUENCY WITHIN LIMIT DETERMINED BY Ta/Tb */
 set_power(table);              /* LOWER VOLTAGE REFERRING TO TABLE */
 do_something();                /* PROCESSING TO BE DONE WITH FIRST PROGRAM.  PROCESSING TIME ESTIMATE IS Ta */

 while (B_is_done)              /* WAIT FOR COMPLETION OF SECOND PROGRAM */
     /* do nothing */;
}

interrupt_from_B()              /* INTERRUPT HANDLING ROUTINE OF FIRST PROGRAM */
{
 restore_power();               /* RETURN VOLTAGE TO ORIGINAL STATE */
 restore_clock_frequency();     /* RETURN CLOCK FREQUENCY TO ORIGINAL STATE */
 B_is_done=1;
}
```

FIG. 6

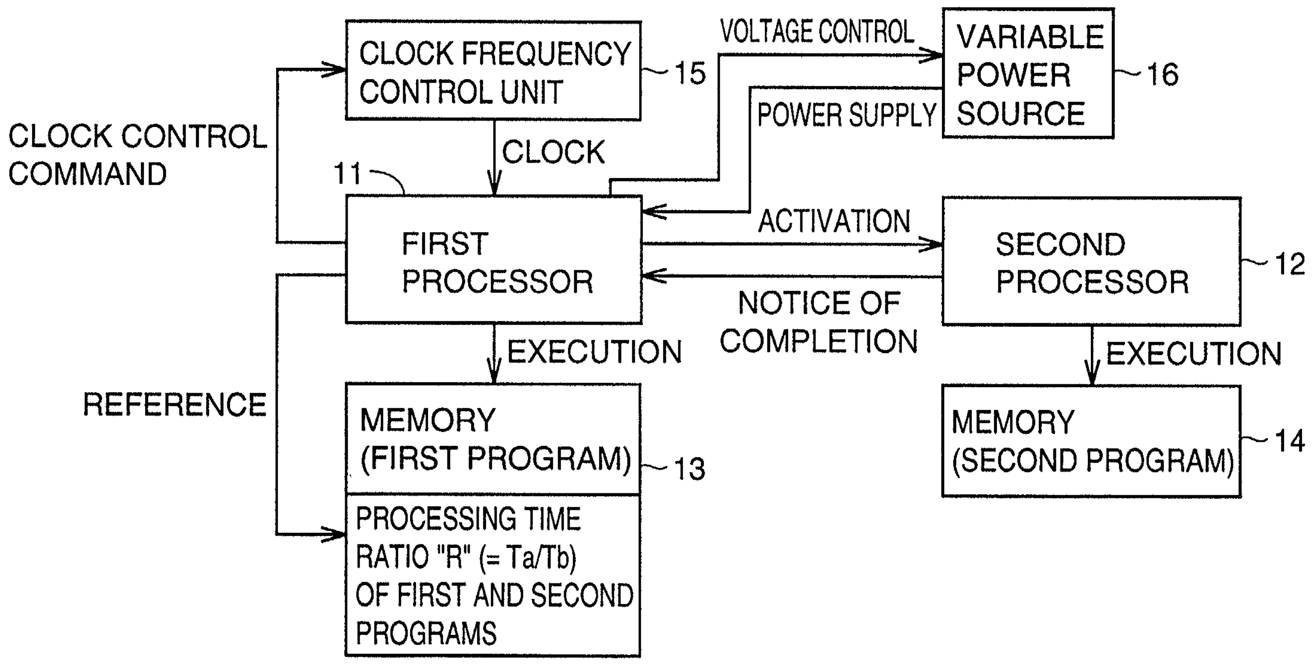

CLOCK FREQUENCY CONTROL UNIT
15
VOLTAGE CONTROL
VARIABLE POWER SOURCE
16
POWER SUPPLY
CLOCK CONTROL COMMAND
CLOCK
11
FIRST PROCESSOR
ACTIVATION
SECOND PROCESSOR
12
NOTICE OF COMPLETION
EXECUTION
EXECUTION
REFERENCE
MEMORY (FIRST PROGRAM)
13
MEMORY (SECOND PROGRAM)
14
PROCESSING TIME RATIO "R" (= Ta/Tb) OF FIRST AND SECOND PROGRAMS

FIG. 7

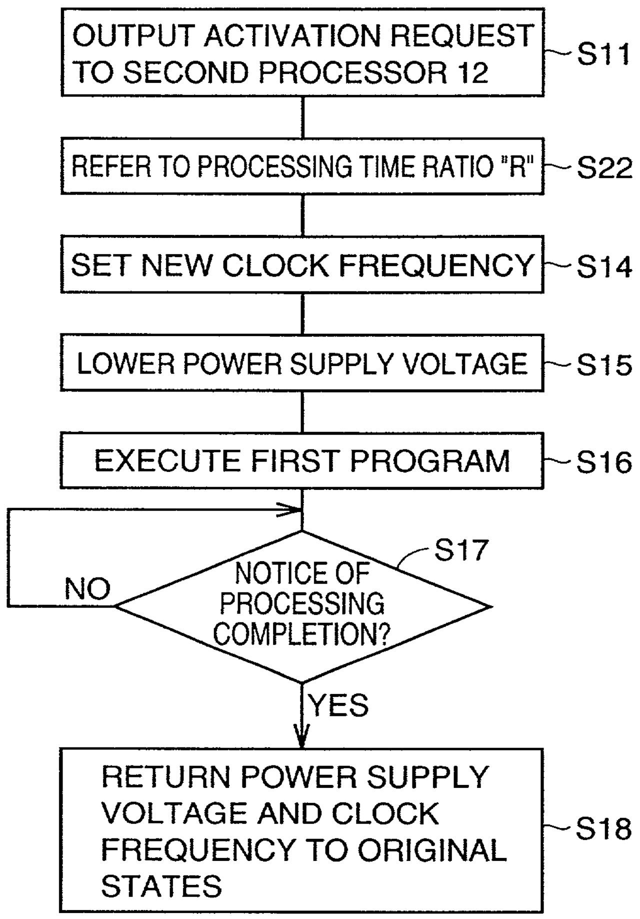
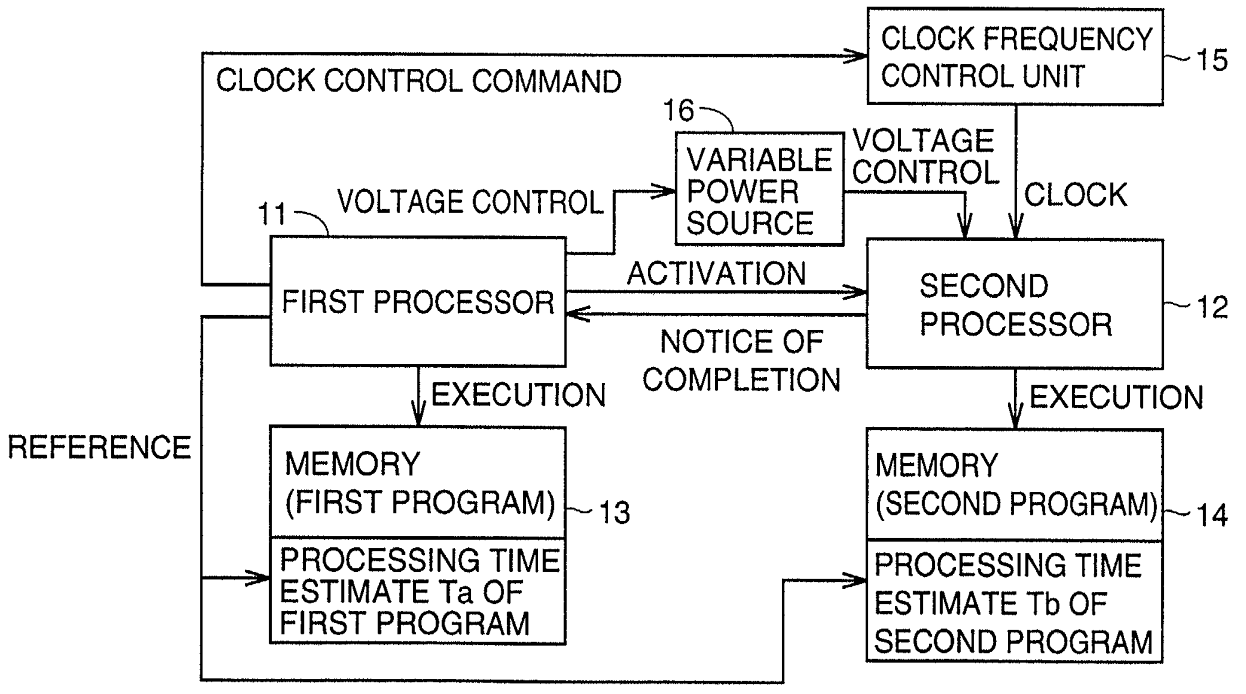
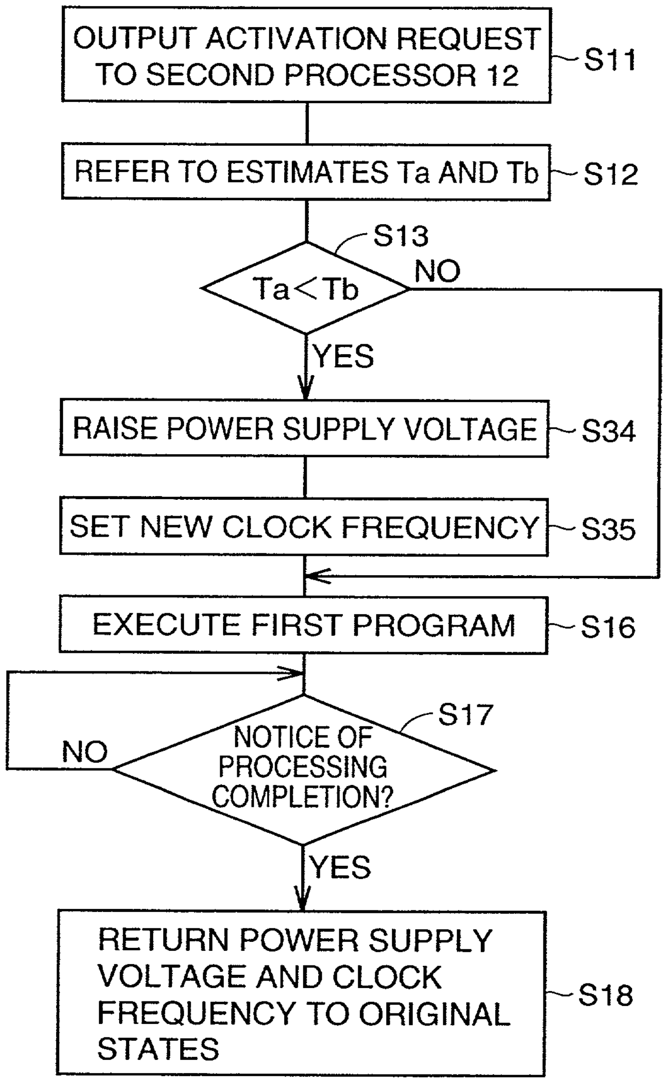

OUTPUT ACTIVATION REQUEST TO SECOND PROCESSOR 12
S11
REFER TO PROCESSING TIME RATIO "R"
S22
SET NEW CLOCK FREQUENCY
S14
LOWER POWER SUPPLY VOLTAGE
S15
EXECUTE FIRST PROGRAM
S16
NOTICE OF PROCESSING COMPLETION?
S17
NO
YES
RETURN POWER SUPPLY VOLTAGE AND CLOCK FREQUENCY TO ORIGINAL STATES
S18

FIG. 8



CLOCK CONTROL COMMAND
CLOCK FREQUENCY CONTROL UNIT
15
16
VARIABLE POWER SOURCE
VOLTAGE CONTROL
CLOCK
11
VOLTAGE CONTROL
FIRST PROCESSOR
ACTIVATION
NOTICE OF COMPLETION
SECOND PROCESSOR
12
EXECUTION
EXECUTION
REFERENCE
MEMORY (FIRST PROGRAM)
13
PROCESSING TIME ESTIMATE Ta OF FIRST PROGRAM
MEMORY (SECOND PROGRAM)
14
PROCESSING TIME ESTIMATE Tb OF SECOND PROGRAM

FIG. 9



OUTPUT ACTIVATION REQUEST TO SECOND PROCESSOR 12
S11
REFER TO ESTIMATES Ta AND Tb
S12
S13
Ta<Tb
NO
YES
RAISE POWER SUPPLY VOLTAGE
S34
SET NEW CLOCK FREQUENCY
S35
EXECUTE FIRST PROGRAM
S16
S17
NO
NOTICE OF PROCESSING COMPLETION?
YES
RETURN POWER SUPPLY VOLTAGE AND CLOCK FREQUENCY TO ORIGINAL STATES
S18

FIG. 10

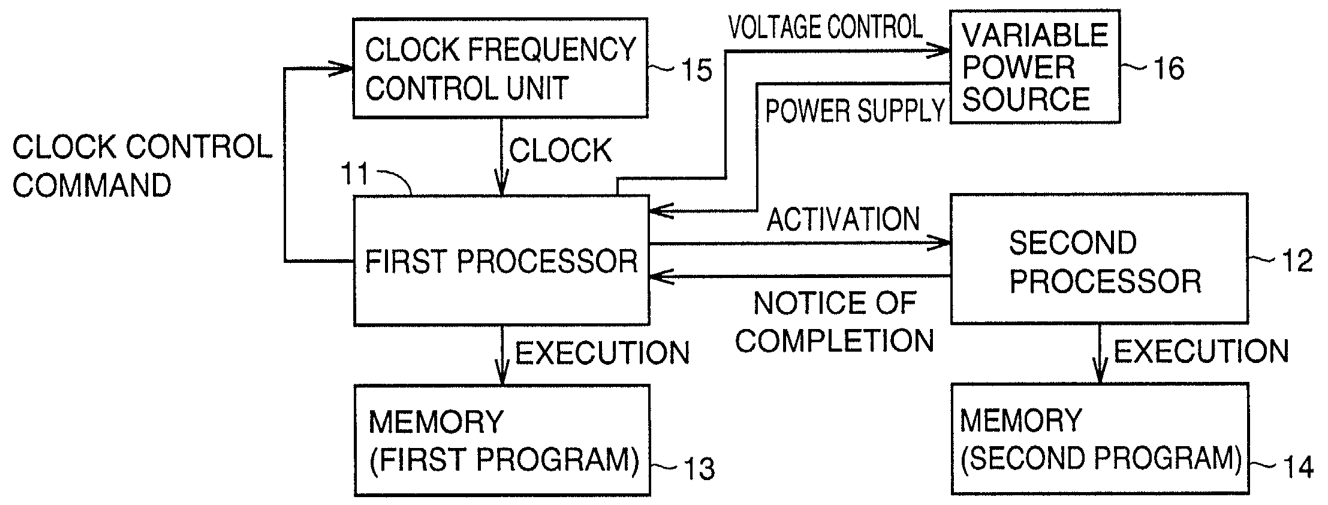

CLOCK CONTROL COMMAND
CLOCK FREQUENCY CONTROL UNIT
15
CLOCK
11
FIRST PROCESSOR
EXECUTION
MEMORY (FIRST PROGRAM)
13
VOLTAGE CONTROL
POWER SUPPLY
VARIABLE POWER SOURCE
16
ACTIVATION
NOTICE OF COMPLETION
SECOND PROCESSOR
12
EXECUTION
MEMORY (SECOND PROGRAM)
14

FIG. 11

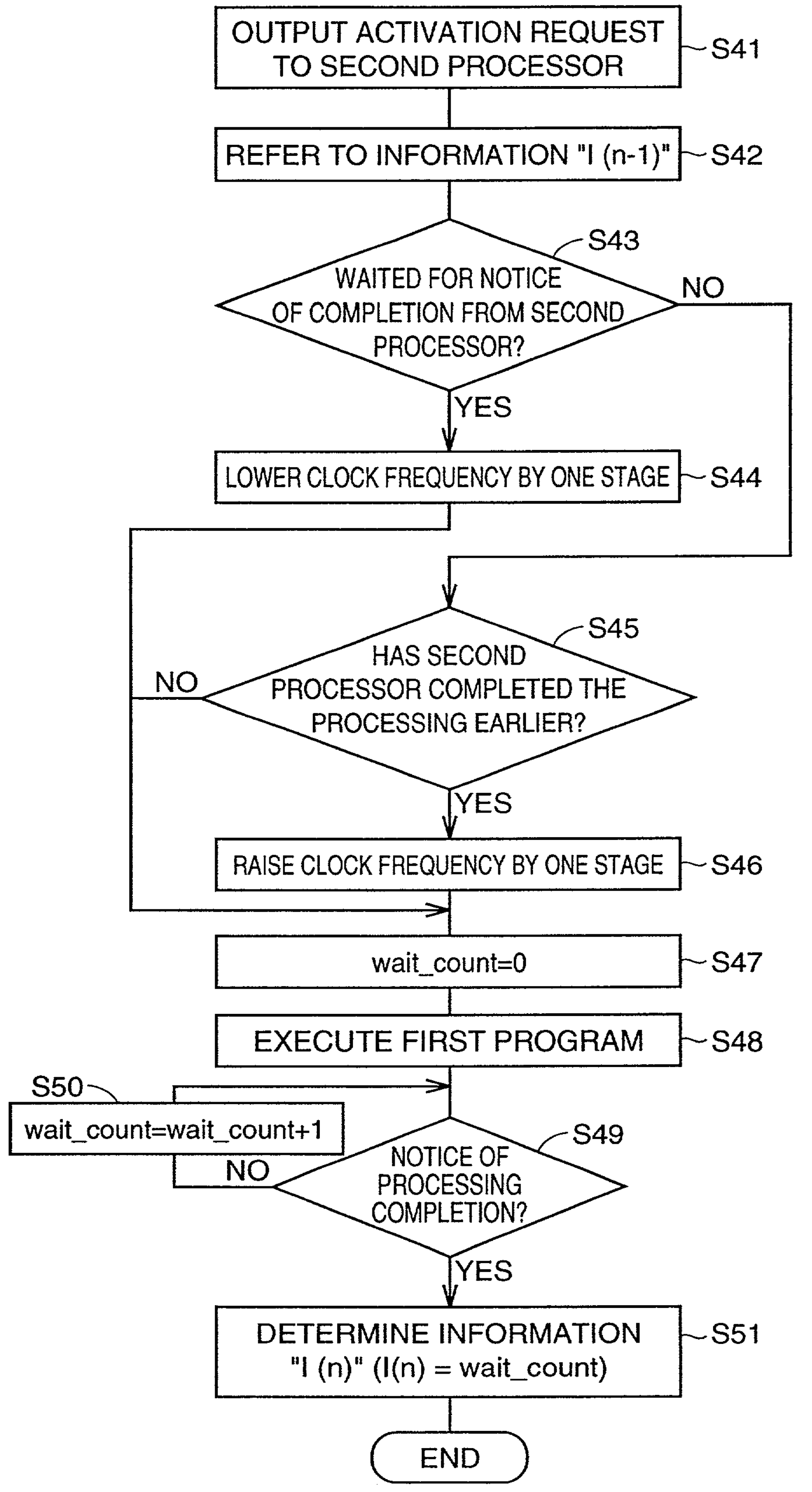
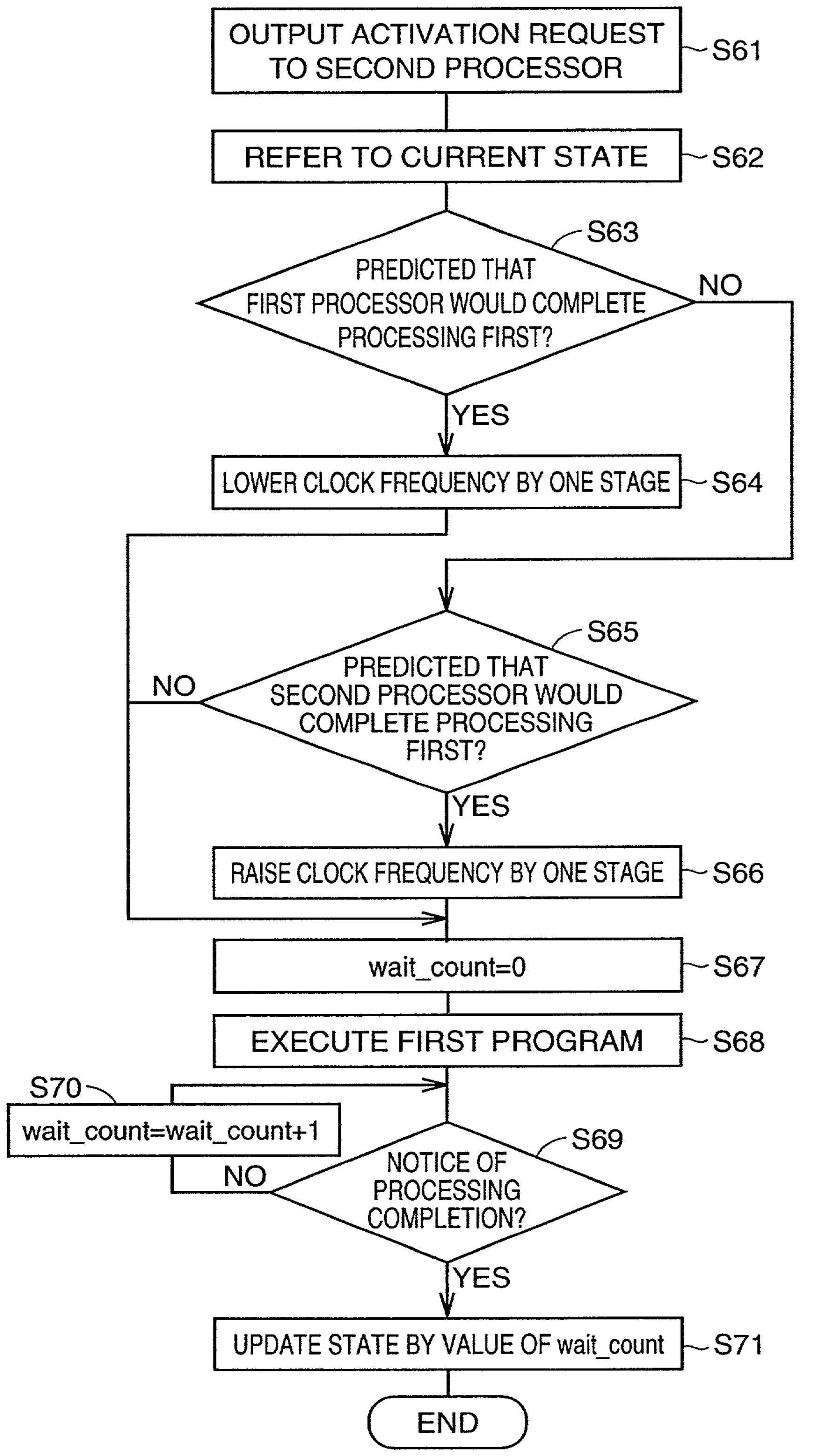

OUTPUT ACTIVATION REQUEST TO SECOND PROCESSOR
S41
REFER TO INFORMATION "I (n-1)"
S42
S43
WAITED FOR NOTICE OF COMPLETION FROM SECOND PROCESSOR?
NO
YES
LOWER CLOCK FREQUENCY BY ONE STAGE
S44
S45
HAS SECOND PROCESSOR COMPLETED THE PROCESSING EARLIER?
NO
YES
RAISE CLOCK FREQUENCY BY ONE STAGE
S46
wait_count=0
S47
EXECUTE FIRST PROGRAM
S48
S50
wait_count=wait_count+1
NO
NOTICE OF PROCESSING COMPLETION?
S49
YES
DETERMINE INFORMATION "I (n)" (I(n) = wait_count)
S51
END

FIG. 12

```
main_procedure()                 /* MAIN PROCESSING UNIT OF FIRST PROGRAM */
{

  B_is_done=0;
  invokeB();                     /* ACTIVATE SECOND PROGRAM */

  if(wait_count>WAIT_LIMIT)      /* IF COMPLETION OF SECOND PROGRAM WAS WAITED IN PREVIOUS PROCESSING, */
      set_clock_faster();        /* LOWER CLOCK FREQUENCY BY ONE STAGE */
  else if(wait_count==0)         /* IF SECOND PROGRAM WAS COMPLETED EARLIER, */
      set_clock_slower();        /* RAISE CLOCK FREQUENCY BY ONE STAGE */
  else                           /* IF FIRST AND SECOND PROGRAMS WERE COMPLETED APPROXIMATE AT THE SAME TIME, */
      /* do nothing */;          /* DO NOTHING AND KEEP CURRENT CLOCK FREQUENCY */
  wait_count=0;

  do_something();                /* PROCESSING TO BE DONE IN FIRST PROGRAM */

  while (B_is_done==0)           /* WAIT FOR COMPLETION OF SECOND PROGRAM */
      wait_count++;
}

interrupt_from_B()               /* INTERRUPT HANDLING ROUTINE OF FIRST PROGRAM */
{
  B_is_done=1;
}
```

FIG. 13



OUTPUT ACTIVATION REQUEST TO SECOND PROCESSOR
S61
REFER TO CURRENT STATE
S62
S63
PREDICTED THAT FIRST PROCESSOR WOULD COMPLETE PROCESSING FIRST?
NO
YES
LOWER CLOCK FREQUENCY BY ONE STAGE
S64
S65
PREDICTED THAT SECOND PROCESSOR WOULD COMPLETE PROCESSING FIRST?
NO
YES
RAISE CLOCK FREQUENCY BY ONE STAGE
S66
wait_count=0
S67
EXECUTE FIRST PROGRAM
S68
S70
wait_count=wait_count+1
NO
NOTICE OF PROCESSING COMPLETION?
S69
YES
UPDATE STATE BY VALUE OF wait_count
S71
END

FIG. 14

```
main_procedure()                /* MAIN PROCESSING UNIT OF FIRST PROGRAM */
{

 B_is_done=0;
 invokeB();                     /* ACTIVATE SECOND PROGRAM */

 PREDICTED VALUE = OBTAIN PREDICTED VALUE FROM CURRENT STATE;
 if (PREDICTED VALUE = "PREDICT THAT PROCESSING OF FIRST PROGRAM WILL BE COMPLETED FIRST")
     set_clock_faster();        /* LOWER CLOCK FREQUENCY BY ONE STAGE */
 else (PREDICTED VALUE = "PREDICT THAT PROCESSING OF SECOND PROGRAM WILL BE COMPLETED FIRST")
     set_clock_slower();        /* RAISE CLOCK FREQUENCY BY ONE STAGE */
 else /* PREDICT THAT FIRST AND SECOND PROGRAMS WILL BE COMPLETED APPROXIMATELY AT THE SAME TIME */
     /* do nothing */;          /* DO NOTHING AND KEEP CURRENT CLOCK FREQUENCY */
 wait_count=0;

 do_something();                /* PROCESSING TO BE DONE IN FIRST PROGRAM */

 while (B_is_done==0)           /* WAIT FOR COMPLETION OF SECOND PROGRAM */
     wait_count++;
 UPDATE STATE BY VALUE OF wait__count;
}

interrupt_from_B()              /* INTERRUPT HANDLING ROUTINE OF FIRST PROGRAM */
{
 B_is_done=1;
}
```

FIG. 15

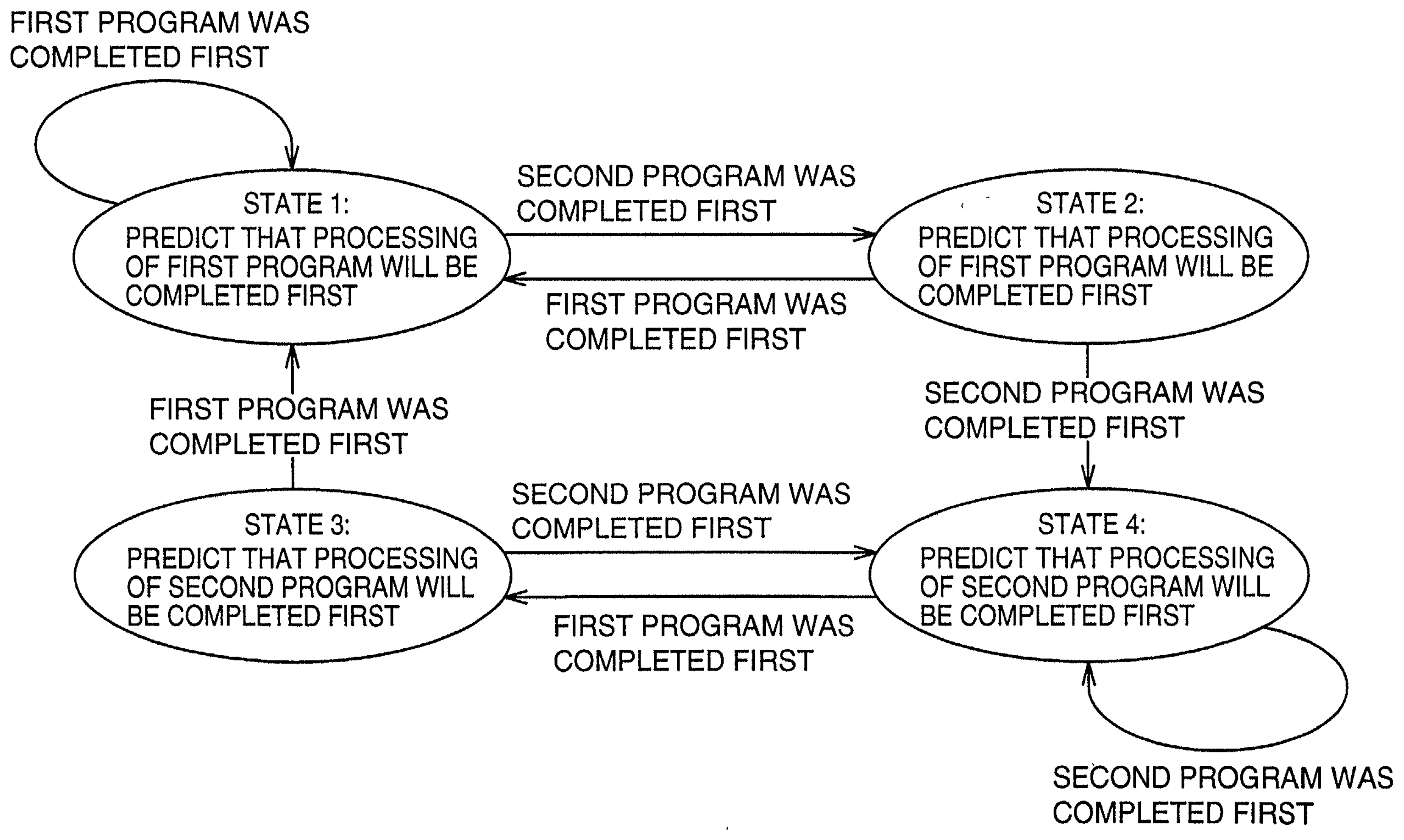

FIRST PROGRAM WAS COMPLETED FIRST
STATE 1: PREDICT THAT PROCESSING OF FIRST PROGRAM WILL BE COMPLETED FIRST
SECOND PROGRAM WAS COMPLETED FIRST
STATE 2: PREDICT THAT PROCESSING OF FIRST PROGRAM WILL BE COMPLETED FIRST
FIRST PROGRAM WAS COMPLETED FIRST
FIRST PROGRAM WAS COMPLETED FIRST
SECOND PROGRAM WAS COMPLETED FIRST
STATE 3: PREDICT THAT PROCESSING OF SECOND PROGRAM WILL BE COMPLETED FIRST
SECOND PROGRAM WAS COMPLETED FIRST
STATE 4: PREDICT THAT PROCESSING OF SECOND PROGRAM WILL BE COMPLETED FIRST
FIRST PROGRAM WAS COMPLETED FIRST
SECOND PROGRAM WAS COMPLETED FIRST

FIG. 16

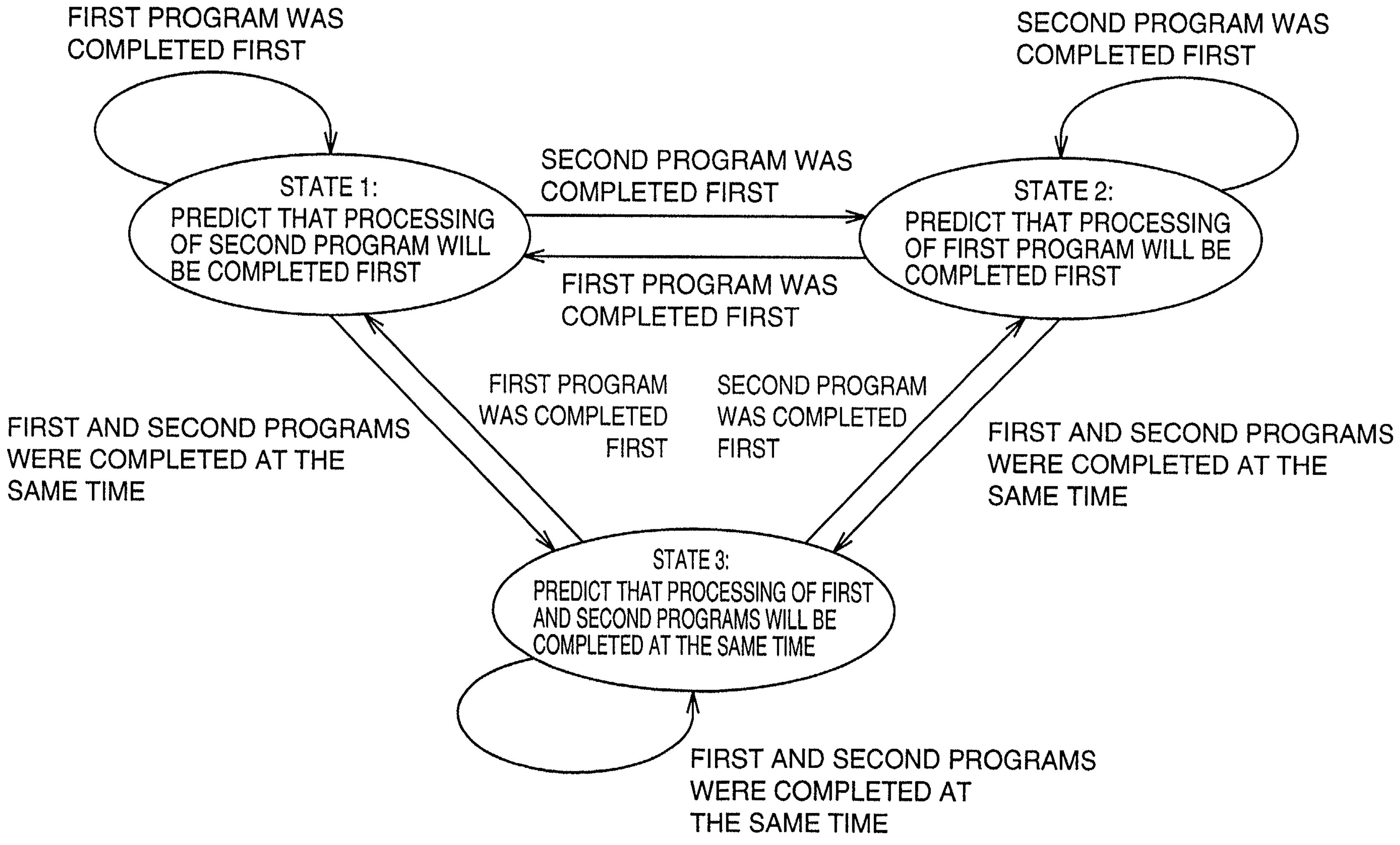
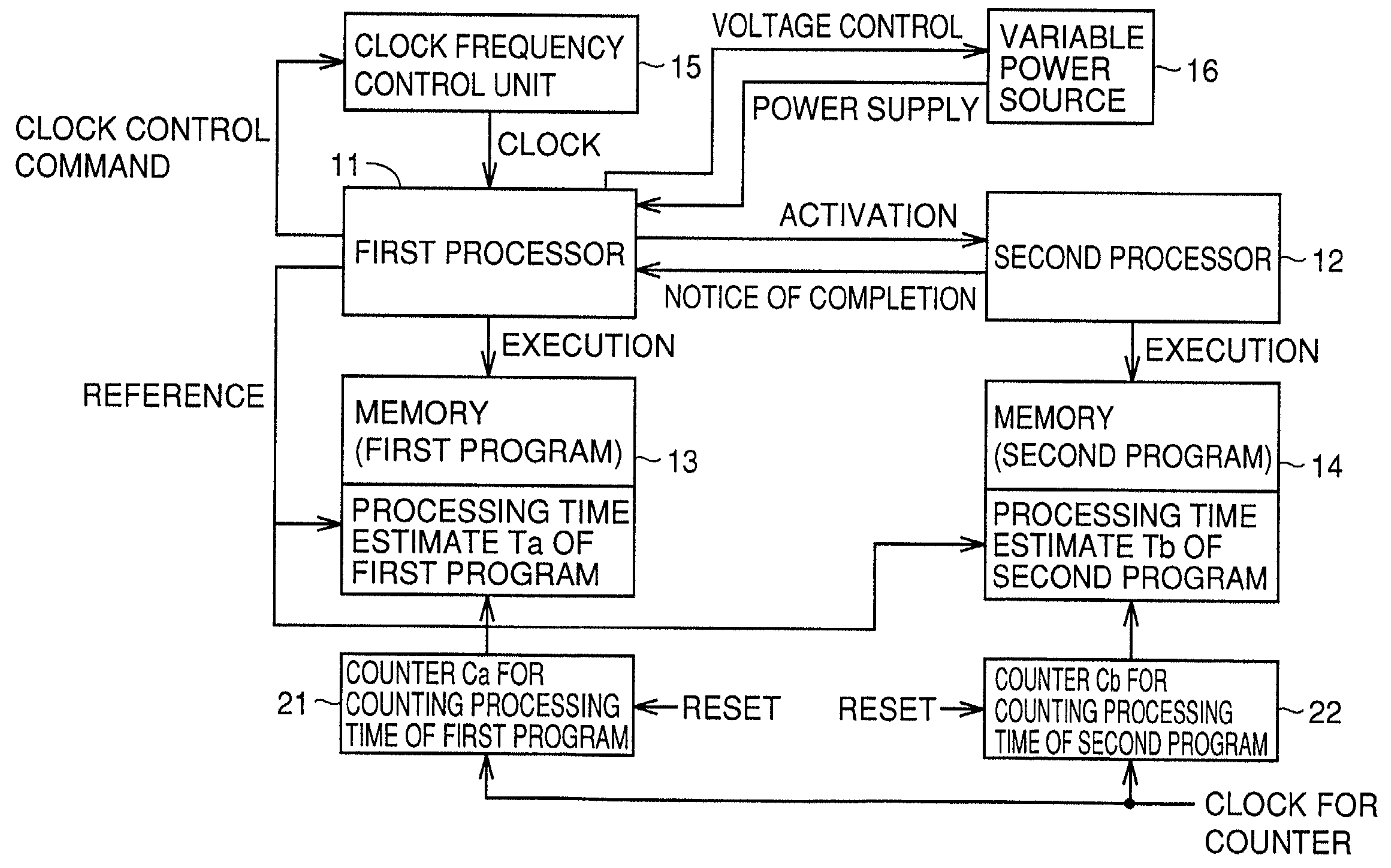

FIRST PROGRAM WAS COMPLETED FIRST
SECOND PROGRAM WAS COMPLETED FIRST
STATE 1: PREDICT THAT PROCESSING OF SECOND PROGRAM WILL BE COMPLETED FIRST
SECOND PROGRAM WAS COMPLETED FIRST
STATE 2: PREDICT THAT PROCESSING OF FIRST PROGRAM WILL BE COMPLETED FIRST
FIRST PROGRAM WAS COMPLETED FIRST
FIRST PROGRAM WAS COMPLETED FIRST
SECOND PROGRAM WAS COMPLETED FIRST
FIRST AND SECOND PROGRAMS WERE COMPLETED AT THE SAME TIME
FIRST AND SECOND PROGRAMS WERE COMPLETED AT THE SAME TIME
STATE 3: PREDICT THAT PROCESSING OF FIRST AND SECOND PROGRAMS WILL BE COMPLETED AT THE SAME TIME
FIRST AND SECOND PROGRAMS WERE COMPLETED AT THE SAME TIME

FIG. 17


CLOCK FREQUENCY CONTROL UNIT
15
VOLTAGE CONTROL
VARIABLE POWER SOURCE
16
POWER SUPPLY
CLOCK CONTROL COMMAND
CLOCK
11
FIRST PROCESSOR
ACTIVATION
SECOND PROCESSOR
12
NOTICE OF COMPLETION
EXECUTION
EXECUTION
REFERENCE
MEMORY (FIRST PROGRAM)
13
MEMORY (SECOND PROGRAM)
14
PROCESSING TIME ESTIMATE Ta OF FIRST PROGRAM
PROCESSING TIME ESTIMATE Tb OF SECOND PROGRAM
21
COUNTER Ca FOR COUNTING PROCESSING TIME OF FIRST PROGRAM
RESET
RESET
COUNTER Cb FOR COUNTING PROCESSING TIME OF SECOND PROGRAM
22
CLOCK FOR COUNTER